POLARITYSPAM: PROPAGATING CONTENT-BASED INFORMATION THROUGH A WEB-GRAPH TO DETECT WEB-SPAM

F. Javier Ortega, José A. Troyano, Fermín L. Cruz
and Carlos G. Vallejo

Department of Languages and Computer Systems
University of Seville
Avda. Reina Mercedes, s/n, 41012, Seville, Spain
{ javierortega; troyano; fcruz; vallejo }@us.es

Abstract. *Spam web pages have become a problem for Information Retrieval systems due to the negative effects that this phenomenon can cause in their results. In this work we tackle the problem of detecting these pages with a propagation algorithm that, taking as input a web graph, chooses a set of spam and not-spam web pages in order to spread their spam likelihood over the rest of the network. Thus we take advantage of the links between pages to obtain a ranking of pages according to their relevance and their spam likelihood. Our intuition consists in giving a high reputation to those pages related to relevant ones, and giving a high spam likelihood to the pages linked to spam web pages. We introduce the novelty of including the content of the web pages in the computation of an a priori estimation of the spam likelihood of the pages, and propagate this information. Our graph-based algorithm computes two scores for each node in the graph. Intuitively, these values represent how bad or good (spam-like or not) is a web page, according to its textual content and its relations in the graph. The experimental results show that our method outperforms other techniques for spam detection.*

Keywords: Information retrieval, Web spam detection, Graph algorithms, PageRank, Web search

1. **Introduction.** Together with the appearance of the first web search engines intended to provide the users with a set of relevant web pages given a user information need or query, many companies came up for the purpose of offering the clients not only the creation of a web site, but also the maximum visibility for them. It was usually done by taking advantage of the mechanisms of the web search engines to rank the web pages according to their relevance in addition to their similarity to the user query. The first approaches to achieve this goal used *keyword stuffing* methods, based on the inclusion of some frequent words in order to get a web page to be highly ranked. Web Spam was firstly known in the literature as *search engine persuasion* [18], due to its goal of making a web page more visible by taking advantage of the search engines. Later, it took the name of *spam* from the similar method that was being used for the massive sending of advertisements and fraudulent or simply annoying messages in newsgroups and e-mails. Web spam mechanisms have evolved from those first attempts becoming more sophisticated, in parallel to search engines counter-measures.

Basically, web spam is a phenomenon where web pages are manipulated for the purpose of obtaining some kind of benefits by illicitly gaining web traffic. Nowadays, spammers use a wide variety of methods intended to make a search engine deliver undesirable results and rank these web pages higher than they would otherwise [21]. There are two basic forms of web spam: Self and Mutual promotion [5]. Self promotion tries to create a web page that gains high relevance for a search engine, mainly based on its content. It can be achieved through many techniques, such as the above mentioned keyword stuffing, in which visible or invisible keywords are inserted in the page, in order to improve its rank for the most common queries. Mutual promotion is based on the cooperation of various sites in order to benefit each other. It usually implies the creation of a wide number of web pages that form a *link-farm*, which is a large number of pages pointing one to another, in order to improve their scores by increasing the number of in-links to them. This method is effective against search engines that employ co-citations between pages as features (e.g., PageRank [23]).

There are several approaches intended to deal with the problem of web spam using different information sources to decide whether a web page is spam or not. Some techniques are based on document classification approaches and try to identify the spam web pages according to some features. Other techniques are focused on the computation of a ranking of web pages, trying to demote the spam web pages in the last positions of the ranking. The intuition behind these methods is that a user who uses a web search engine will take mainly the top ranked documents retrieved by the system, so the goal is to avoid the appearance of spam web pages in those first positions.

In this work, we introduce a novel method that computes a ranking of web pages according to their spam likelihood. It is based on a propagation algorithm that spreads over the graph the information about the spam or not-spam likelihood of the web pages. This information is computed regarding the textual content of the pages. Unlike other web spam detection systems, our method does not need any human intervention in the process. In spite of this, it achieves very good results in the task.

The organization of the rest of the paper is as follows. In the next section, we discuss other works that tackle the problem of web spam detection from different points of view. In Section 3, we introduce the intuition behind our approach, and explain the components of our method: the set of content-based metrics, and the way in which these heuristics are used in the creation of the graph model. The experimental design and results are shown in Section 4. Finally, we remark on our conclusions concerning the present work, and talk about some ideas for future works.

2. **Related Work.** As mentioned above, one of the most common methods for web spam detection are content-based techniques. These methods are focused on determining whether a page is spam or not according to its textual content. Mishne et al. [19] propose the comparison of language models to classify texts as spam or non-spam. In [15], Kolari et al. present a machine learning technique based on SVM, taking as features different heuristics such as the anchor text of the links in a web page, the tokenized URL of the page, or the meta-tag text. In [22], several spam detection metrics are proposed. They compare the values of these content-based metrics for spam and not-spam web pages, and discuss the discrimination ability of each metric to detect spam. Some of the proposed heuristics are the number of words in the title of a web page, the average length of words, the amount of invisible content, the compression rate of the web pages, the fraction of anchor text with respect to the total amount of text in a page, etc. Other works focus their attention in the selection of features that optimize the performance of the spam detection systems [17].

On the other hand, link-based techniques focus on the structure of the graph made up of the web pages and the hyperlinks among them. These methods study the relations of the pages in the web graph, aiming to detect the link-farms of spam web pages (see Figure 1). The basic assumption to deal with link-farms is that similar objects are related to similar objects in the web graph [13]. In the context of web spam, it means that non-spam web pages are frequently related to other non-spam web pages, and vice versa. Link-based methods are intended to deal with the mutual promotion mechanisms.

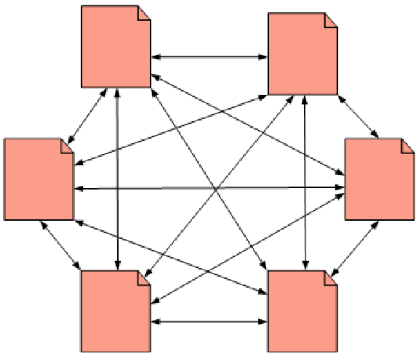

FIGURE 1. Graphical representation of a link-farm of web pages: a set of web pages with a high number of links between them

The key assumption in [2] is that supporters of a non-spam page should not be overly dependent on one another. In other words, if the supporters of a web page have a large numbers of links between them, they likely form a *link-farm* and could be spam web pages. An example of suspicion is the case of a page that receives its PageRank from a large number of very low ranked pages. The proposed algorithm obtains the supporters of each page, then studies the distribution of their PageRank scores in order to compute a PageRank biased according to a vector of penalizations.

Truncated PageRank [1] tackles also the problem of the link-farms. It penalizes pages that obtain a large share of their PageRank from the nearest neighbors, avoiding the effect of the supporters that are topologically very close to a given node.

TrustRank [9] is based on the idea that a high PageRank score held on a huge amount of links from pages with low PageRank, is suspicious of being spam. It means that a node with high PageRank and no relations with others pages with high PageRank is likely to be a spam web page. They obtain an estimator for this metric by calculating the *estimated non-spam mass*, that is the amount of PageRank received from a set of (hand-picked) trusted pages. TrustRank takes as input the web graph, and a set of hand-picked non-spam web pages. In contrast, [16] proposes an algorithm with the same idea, but taking as input a set of spam web pages. This technique, called Anti-TrustRank, computes the *estimated spam mass* for each node. Both methods need some human intervention in order to choose the set of sources for the ranking algorithms. This fact limits the number of sources that can be taken into account by both methods.

In [25], Wu et al. propose an approach based on trust and distrust propagation. This work consists in an algorithm that computes two scores for each node in the graph, indicating the levels of trust and distrust of a page. The process starts from two source sets, trustworthy and spam pages, respectively.

A system that combines content-based and link-based features is proposed in [4]. They discuss three methods to include features related to the web graph topology into a classifier. Some well-known algorithms are used in this work, such us PageRank or the above mentioned TrustRank and Truncated PageRank. Similar ideas has been applied to enhance the performance of other information retrieval systems [10, 24].

The impact of spam in information retrieval systems, and the effects of some anti-spam filters are studied in [5]. They use three filters in this work, and a naive Bayes classifier to combine all of them. The first filter is a classifier built from a labeled corpus with spam and non-spam pages. The second one consists of a set of documents retrieved by some of the most popular queries to a web search engine. And finally a set of documents extracted from the Open Directory Project[1]. They show the improvements achieved in some of the systems participants in TREC 2009[2] applying a spam detection technique.

3. **PolaritySpam.** Both link-based and content-based techniques have been shown to be reliable methods for web spam detection, although they present some vulnerabilities that can be easily exploited by spammers. In this section we present PolaritySpam as our proposal for web spam detection. It is a method that includes concepts from link-based and content-based techniques, combining the strongest points of them.

PolaritySpam is based on similar ideas as PolarityRank [7], a ranking algorithm over graphs with positive and negative weights in its edges. This algorithm propagates the information of a set of positive and negative seeds through the graph, obtaining the polarity of the rest of nodes. The intuition behind PolaritySpam is the propagation of spam and not-spam likelihood of web pages over the web graph. It relies on the assumption that similar objects are related to similar objects in the web graph [13]. So every web page related to a spam-like web page must be considered as spam, and vice versa. In other words, each page propagates its spam likelihood to their neighbors, so every web page will receive good or bad reputation regarding to the pages that are pointing to them.

PolaritySpam consists in a propagation algorithm that computes two scores for each node: a positive score representing the authority of a web page, and a negative score which represents the spam likelihood of a page. The difference between both scores is taken into account in order to build a ranking of web pages. Intuitively, this value represents the overall relevance of a web page. In this way, web pages with high negative scores are demoted in the final ranking, because they are likely to be spam.

Following this reasoning, we need some kind of a priori knowledge about the web pages in order to determine their spam likelihood before starting the propagation. At this point, we propose the use of some heuristics to extract this information from the textual content of the pages. These metrics are the basis of an automatic process intended to obtain an a priori score for each page, weighting their spam likelihood. Then, we compute a *spam-biased* random-walk algorithm to propagate this a priori information through the web graph. We take advantage of the links in the graph to spread the content-based information over the network. In this way, we can promote those pages related to relevant web pages, because they will receive high positive scores. On the contrary, those web pages related to spam-like pages are demoted in the ranking because they will receive high negative scores.

In Figure 2, we show the general schema of our system. It consists of three components: the A-priori Information Extractor that processes the textual content of the web pages in order to obtain a score representing their spam likelihood; the Selector of Sources that automatically selects two sets of spam and not-spam like web pages, respectively, regarding the a priori information previously computed; and the Propagation Algorithm that processes the web graph, propagating the a priori information of the sources according to the links between the web pages.

In this work, we propose three variations for our system. Each method automatically selects a number of pages (namely *sources*) to spread their positive or negative influence to the rest of the network. Then, they specify the weight of each source to be propagated. Since we compute two scores for each page, we need two sets of sources, each of them

[1] http://www.dmoz.org

[2] http://trec.nist.gov

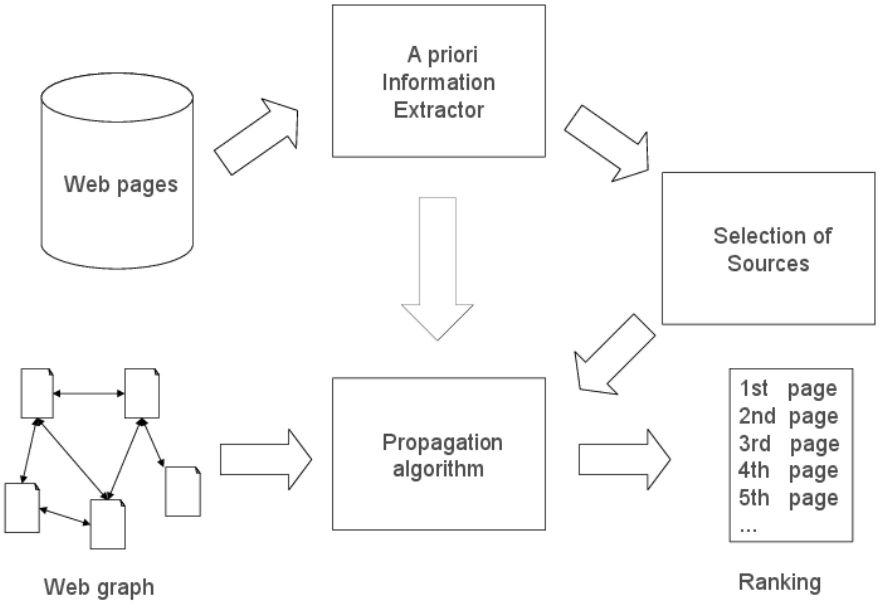

Figure 2. Structure of our spam detection system

intended to reinforce the positive or negative relevance of each type of web pages in the graph. Thus, the first set of sources must contain a group of non-spam pages, and the second one consists in a group of spam pages.

Isolated pages, i.e., pages without any in-links or out-links are also a problem for random-walk algorithms. Indeed, despite not being able to obtain any feedback from their neighbors, the textual content itself provides useful information to obtain a spam likelihood score for these pages.

In the remainder of this section, we discuss the content-based metrics and their role in our system (Section 3.1), the automatic methods to obtain the sets of sources (Section 3.2), and finally the algorithm proposed to propagate the content-based information of the web pages over the network (Section 3.3).

3.1. **Extracting a priori information.** The intuition behind the use of the content-based metrics in conjunction with a random-walk algorithm lies on the propagation of the spam likelihood of the web pages over the graph. We propose the use of the information provided by the textual content of the web pages as spam likelihood indicators. In this way, we provide some a priori information to our system in order to begin the propagation graph-based algorithm. The aim behind this idea is to increase the ranking of the good web pages and penalize the bad ones, regarding the content of the pages in addition to the link structure of the graph.

The content-based metrics that we use in the experiments of this work have been chosen according to their discrimination capacity, distinguishing between spam and not-spam web pages, as identified by [22]. Another important factor to select these metrics is their computational complexity. Following these criteria, we have implemented two heuristics:

- **Compressibility**: is defined as the fraction of the sizes of a web page, x, before and after being compressed.

$$Compressibility(v_j) = \frac{GZIPSize(v_j)}{TotalSize(v_j)}$$

  A web page with a very high compressibility value, is likely to be a spam. This heuristic is intended to detect repeated content or words in a web, because more redundant content leads to a higher compression ratio.
- **Average length of words**: non-spam web pages have a bell-shaped distribution of average word lengths, while malicious pages have much higher values of this metric. Hence, this heuristic penalizes the use of word stuffing mechanisms.

In the next section, we explain the different ways of including these heuristics into our system.

3.2. **Selection of sources.** PolaritySpam uses the content-based heuristics to automatically assign a priori scores to some specific pages (henceforth called *sources*), regarding their spam or not-spam likelihood. A given page will be demoted or promoted in the final ranking according to its relations with these sources. We use sources sets to ensure that negative scores of negative pages will be propagated over the graph, and analogous for the positive sources. The sources sets are represented in our approach by two spam-biased vectors, e^+ and e^-. The vectors contain the spam and non-spam likelihoods of the web pages taken as sources in our algorithm, giving higher positive or negative scores to those nodes that have higher e^+ or e^- (see Equations (4) and (5)).

In this section, we introduce three different ways to characterize the spam-biased vectors, given the heuristics of each web page.

3.2.1. *Most spamy/not-spamy sources (S-NS).* This first method chooses the N most spam-like and not-spam like pages in the graph as sources, according to their metrics. Formally, given a page v_j, let $M(v_j)$ be a vector with the content-based metrics for v_j. The spam likelihood of v_j will be determined by the norm of $M(v_j)$, as shown in Equation (1):

$$Spaminess(A) = \sqrt{\sum_{h \in M(v_j)} h^2} \tag{1}$$

where A is a web page, and h represents the heuristics which $M(v_j)$ contains.

In this way, we take the N nodes with highest *Spaminess* as negative sources, and the N nodes with lowest *Spaminess* as positive sources. The spam-biased vectors can be defined as follows:

$$e_i^+ = \begin{cases} 1/N & \text{if } i \in S^+ \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where N is a parameter that specifies the number of sources that will be taken. S^+ is the set of the N nodes with lowest Spaminess in the graph. The formula is analogous for vector e^-.

3.2.2. *Content-based weighted spamy/not-spamy sources (CS-NS).* Following the previous schema, we can take advantage of the content-based metrics by including the actual scores directly in the computation of the weights of the sources, as shown in Equation (3):

$$e_i^+ = \begin{cases} \frac{Spaminess(i)}{\sum_{j \in S^+} Spaminess(j)} & \text{if } i \in S^+ \\ 0 & \text{otherwise} \end{cases}$$

$$e_i^- = \begin{cases} \frac{Spaminess(i)}{\sum_{j \in S^-} Spaminess(j)} & \text{if } i \in S^- \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $Spaminess(i)$ is the norm of the vector built from the metrics of the page i (see Equation (1)).

3.2.3. *Content-based graph sources (C-GS).* The last method consists in applying the previous formula to every nodes in the graph. We can rely on the thresholds proposed in the study in [22], and use them to determine whether a page must be a negative or a positive source. The thresholds for the metrics considered in the present work are shown in Table 1. Given a page, if one of its metrics is above the corresponding threshold, we include the page in the set of negative sources, and in other case it will be taken as a positive source. Once the sets of nodes have been defined, we apply the same formulas shown in Equation (3).

Table 1. Thresholds for the content-based metrics

Heuristics	Threshold
Compressibility	6.0
Average length of words	9.0

3.3. **Propagation algorithm.** As mentioned above, we propose a propagation algorithm in order to spread over the graph the information extracted by the content-based metrics, and to compute a ranking of the web pages according to their relevance. This algorithm is intended to demote the spam web pages in the overall ranking by computing two scores for each page, PR^+ and PR^-. Given a page A, it is desirable that its positive score, PR^+, depends on the good pages pointing to A, and analogous for the negative score, PR^-. In other words, we want the spam web pages to propagate their negative scores to their neighbors, and the positive pages do the same with their positive scores. With this aim, two vectors, e^+ and e^-, are built based on a set of content-based metrics from each page. These spam-biased vectors are used in the computation of our random-walk algorithm, representing the non-spam and the spam likelihoods of a page, respectively. They can be thought of a reinforcement for the positive and negative scores of each page. Having said that, the proposed scores are obtained as shown in Equations (4) and (5) below:

$$PR^+(v_i) = (1-d)e_i^+ + d \sum_{j \in In(v_i)} \frac{PR^+(v_j)}{|Out(v_j)|} \tag{4}$$

$$PR^-(v_i) = (1-d)e_i^- + d \sum_{j \in In(v_i)} \frac{PR^-(v_j)}{|Out(v_j)|} \tag{5}$$

where v_i is a node of the graph (a web page), $In(v_j)$ is the set of nodes pointing to v_j, and $Out(v_j)$ is the set of nodes which v_j points to. Both scores, PR^+ and PR^-, are obtained with a PageRank-like algorithm. The algorithm iterates over the nodes in the graph, applying Equations (4) and (5). This process is performed until the maximum difference between the scores in one iteration and the previous one, is lower than a given threshold. This algorithm has the same time complexity as the original one.

Once the propagation algorithm is finished, a ranking of web pages is built regarding the difference between PR^+ and PR^- for each node, as shown in Equation (6).

$$score(v_i) = PR^+(v_i) - PR^-(v_i) \tag{6}$$

4. **Experiments.** In this section, we show the experimental design defined to show the performance of PolaritySpam, as well as the dataset used and the results obtained. We also detail the values of the parameters for each set of experiments, and the different variants proposed in this work.

The aim of the experiments is to show the performance of PolaritySpam in terms of the demotion of spam web pages in the resulting ranking, in order to asses its usefulness in the web spam detection task. The results of PolaritySpam are compared to a state-of-art web spam detection technique, TrustRank [9], which achieves very good results in this task.

Since PolaritySpam does not classify the web pages between spam or non-spam, it does not make sense to perform an evaluation in terms of classification accuracy. On the other hand, we use in our experiments the same evaluation method followed in other works on the application of graph-based algorithms to the spam detection task [2, 9]. This kind of spam detection techniques are based on the fact that a user usually looks only at the top ranked documents of a result set, ignoring the rest of lower-ranked documents. So the goal of ranking algorithms in the web spam detection task is to demote the spam web pages in such way that they became less visible to the users due to their low rankings. Therefore, the evaluation methods applied in this work take into account the positions obtained by the spam web pages in the rankings computed by the implemented techniques, in order to measure their performance.

The organization of the rest of this section is as follows. The dataset used in the experiments is presented in Section 4.1. The evaluation methods are explained in Sections 4.2 and 4.3. In Section 4.4, we present the TrustRank algorithm, taken as baseline in the evaluation. Finally, we show the experimental results for all the techniques in Section 4.5, comparing the results of the techniques to the TrustRank algorithm using some useful evaluation metrics.

4.1. **Dataset.** The corpus used in the experiments is the WEBSPAM-UK2006 Dataset [3], a huge collection of web pages specifically crawled for researching on spam detection. It contains more than 98 million pages. The collection is based on a crawl of the .uk domain performed in May 2006. It was collected by the Laboratory of Web Algorithmics, Università degli Studi di Milano, with the support of the DELIS EU-FET research project. The collection was labeled by a group of volunteers and/or by domain-specific patterns such as .gov.uk or .ac.uk. Of the 11,402 hosts in UK2006 dataset, 7,423 are labeled as spam or non-spam. For the evaluation purposes, we have considered as spam any web page that belongs to a host labeled as spam. There are about 10 million spam web pages in the collection.

4.2. **Evaluation with PR-buckets.** Since the position of spam web pages in the ranking of documents is the key of our spam detection technique, we focus on this feature in order to evaluate the performance of our approaches. In this section, we explain the PR-buckets method, introduced in [9] to evaluate their proposal. The aim of this method is to easily get a qualitative evaluation by determining the number of spam web pages detected mainly in the highest positions of the ranking of web pages, that are the most pernicious for the performance of a web search engine.

This evaluation method is implemented as follows. First, a list of pages is generated in decreasing order of their PageRank score. This list is segmented into 20 buckets, in such way that each of the buckets contains a different number of sites, with scores summing up to 5% of the total PageRank score. The sizes of these PR-buckets are taken to build a similar set of buckets using the rankings computed by each web spam detection method. In this way, given N_1, the size of the first PR-bucket, we take the N_1 top ranked web pages for each technique in order to build the first bucket corresponding to each result set, and so forth.

The number of spam web pages per bucket is our evaluation metric. It is obtained by counting the number of pages in each bucket that are labeled as "spam" in the dataset. The aim of a spam detection technique is to avoid spam web pages into the first buckets (top positions of the ranking), demoting these pages in order to put them into the last buckets.

4.3. **Evaluation with nDCG.** Another method intended to evaluate the results of ranking algorithms is the *Normalized Discounted Cumulative Gain* [20] (*nDCG*), a well-known metric widely used in Information Retrieval for the evaluation of a set of retrieved documents given a query. This metric measures the quality of a ranking of documents, penalizing the appearance of top-ranked irrelevant documents in the results. It is based on the Discounted Cumulative Gain (DCG), computed as follows:

$$DCG = Relevance_1 + \sum_{i=2}^{N} \frac{Relevance_i}{\log i}$$

where N is the total number of documents in the collection, and $Relevance_i = 1$ if document in the i-th position is relevant (in our case, a relevant document is a not spam web page), and zero in other case. As we can see, the higher are the positions of irrelevant documents in the ranking, the lower is the DCG score of the ranking. $nDCG$ is obtained by normalizing the previous score using the $IDCG$, that is the best DCG that can be achieved with the given dataset:

$$IDCG = \sum_{i=1}^{|Relevance|} \frac{1}{\log i}$$

The normalized score is obtained as follows:

$$nDCG = \frac{DCG}{IDCG}$$

where $0 \leq nDCG \leq 1$, corresponding $nDCG = 1$ to the perfect system that gets all the irrelevant documents ranked in the last positions of the ranking. So, with $nDCG$ we can evaluate the performance of each technique in terms of the demotion of spam web pages in the ranking.

4.4. **Baseline: TrustRank.** TrustRank algorithm [9] is a link-based algorithm that takes as input the web graph and a set of non-spam web pages, chosen in a semi-supervised way, that are the sources for the algorithm. The output is a ranking of web pages according to their relevance, in which the spam web pages are demoted. TrustRank computes a score for each web page, similarly to PageRank, using the source set to include a bias in the random-walk algorithm.

In order to build the source set, they propose an *inverse PageRank* algorithm, that computes a ranking of web pages, but taking into account the outlinks of the web pages instead of their inlinks. Once the inverse ranking has been built, they choose by hand the N top-ranked non-spam web pages from that ranking. In this way, they try to take as sources the N good pages that reach as many nodes as possible, trying to maximize the propagation of the information from these sources. The disadvantage of this method is that human intervention is required, so it is expensive to consider a high number of web pages as sources due to the manual effort needed.

We test this technique with the UK2006 dataset, and it achieves very good results. The results shown in Figure 3 have been obtained by performing 20 iterations of the algorithm with a damping factor of 0.85, as suggested in [9]. We have manually built a source set with 534 non-spam web pages. We can see in the chart that there are less than 3% of spam web pages into the two first buckets. However, more than the 10% of pages in the third bucket are spam.

4.5. **Experimental results.** In this section, we show the results obtained by our approach and its three methods for the selection of sources, comparing their results with the TrustRank algorithm. We have used the same values for the parameters of the propagation algorithm that the ones in previous section: a damping factor of 0.85 and executing 20 iterations of each propagation algorithm.

For the S-NS and CS-NS methods we have selected the 5% of the most positive and negative nodes as the positive and negative source sets, respectively. The results for the first ten buckets are shown in Figure 3, where TR corresponds with TrustRank algorithm; S-NS (Spamy/Not-spamy Sources) is our first approach, that takes the N most positive and negative pages as sources and assigns a weight of $1/N$ to each of them; CS-NS (Content-based weighted Spamy/Not-spamy Sources) is our second approach, based on the previous one, but setting the weights of the sources according to the content-based metrics; and finally, C-GS (Content-based Graph Sources) corresponds with our third approach that takes every node as a source for the random-walk algorithm.

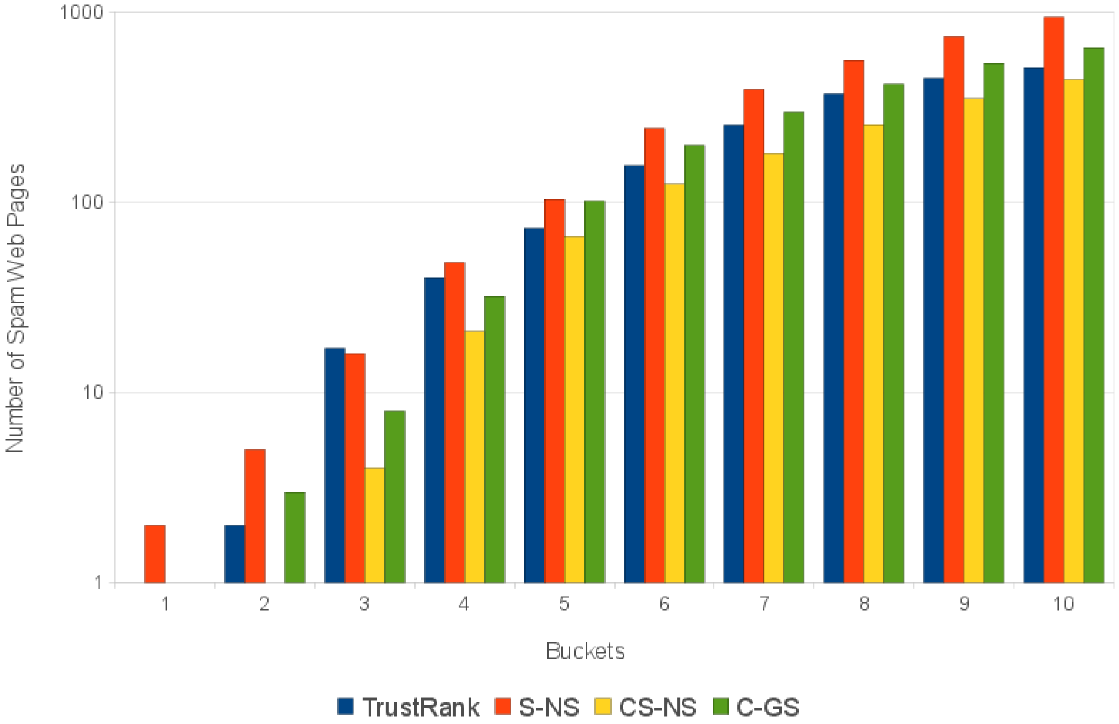

FIGURE 3. Number of spam web pages per bucket for each technique in the first ten buckets

In Figure 3, we can see that TrustRank performs very well, demoting a high number of spam web pages in the ranking, and allowing only a few of them to appear in the top positions of the ranking. Our techniques present a really good behavior as well, avoiding the spam web pages in the first bucket (except for S-NS). CS-NS shows the best results of all the techniques, successfully avoiding the spam web pages in the first and second buckets, and achieving a very low number of errors in the rest of them. Our third approach, C-GS, also outperforms the baseline, showing a really good performance in these first buckets.

In Table 2, we show the number of spam web pages per bucket. The first column represents the buckets of pages and the second one contains the total number of web pages from the first bucket to the current one, inclusive. The rest of the columns shows the accumulated number of spam web pages for each technique, that is the total number of spam web pages from the first bucket to the current one, inclusive.

Table 2. Accumulated number of spam web pages for each method: TrustRank (TR), Spamy/Not-spamy Sources (S-NS), Spamy/Not-spamy Sources with metric-based weights (CS-NS) and Content-based Graph Sources (C-GS). The best results are shown in bold.

Buckets	Pages	TR	S-NS	CS-NS	C-GS
1	14	**0**	2	**0**	**0**
2	68	2	5	**1**	3
3	212	17	16	**4**	8
4	649	40	48	**21**	32
5	1719	73	104	**66**	101
6	3849	155	244	**124**	199
7	6513	254	392	**180**	297
8	9291	371	557	**255**	416
9	12102	448	742	**350**	537
10	14914	511	937	**440**	650
11	17726	653	1112	**543**	781
12	20538	860	1292	**543**	957
13	23350	942	1460	**698**	1103
14	26162	1237	1885	**888**	1299
15	29673	1532	2310	**1060**	1495
16	38002	1827	2735	**1615**	1992
17	16103632	1554162	**693105**	940855	1810597
18	44043924	4657553	**3221681**	3383010	4589582
19	71984216	8138745	**6607552**	7379871	8080963
20	98812333	**10181905**	**10181905**	**10181905**	**10181905**

We can see that CS-NS achieves the best results outperforming TrustRank. In contrast, the first approach does not improve the TrustRank algorithm. This fact, in addition to the results of CS-NS, highlights the relevance of including the content-based metrics in the weights of the sources, and also gives us the idea of the need to improve the mechanism for the selection of the sources. Finally, the C-GS method also presents good results, with only 32 spam web pages in the first 649 pages, outperforming TrustRank in those first buckets as well.

In Figure 4, we show a graphical comparison of the performance of all the techniques. For each bucket b, we compute the proportion of good pages present in the ranking up to that bucket, according to the formula:

$$Precision(bucket_b) = \frac{\sum_{\forall bucket_i <= bucket_b} NumberOfGoodPages(bucket_i)}{\sum_{\forall bucket_i <= bucket_b} NumberOfPages(bucket_i)} \qquad (7)$$

This metric highlights the evolution of the performance of each technique as we advance in the ranking of web pages. We focus our attention again in the first buckets where we try to avoid the appearance of spam web pages, although we plot in Figure 4 the precision of each technique for all the buckets to show the global behavior of all the methods.

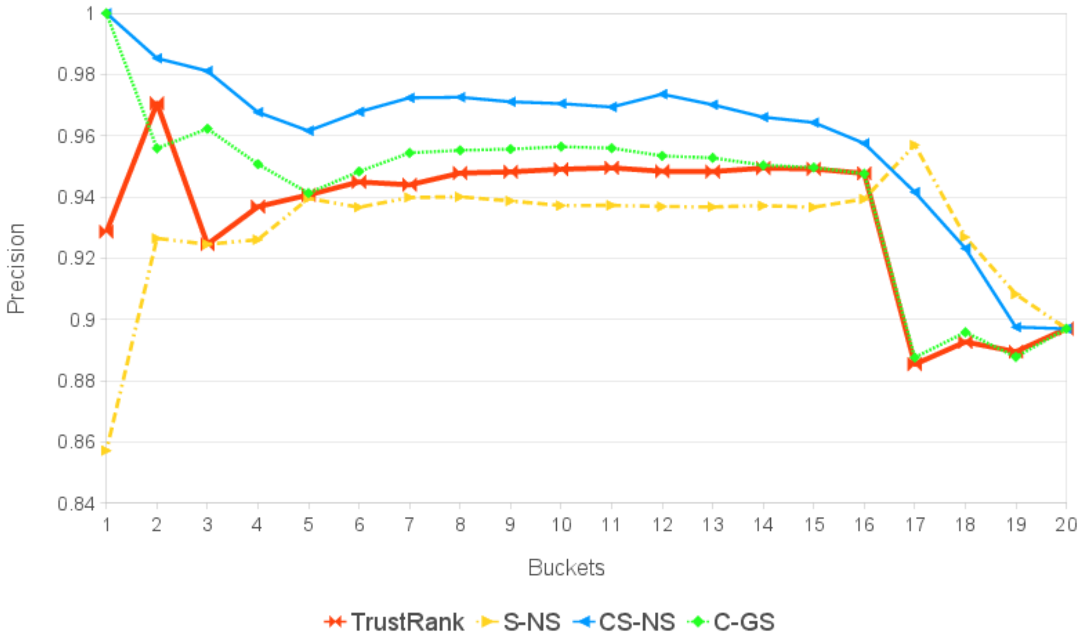

FIGURE 4. Precision computed for each bucket as shown in Equation (7)

In Figure 4, we can see that our first approach, S-NS, under-performs the TrustRank algorithm in the first buckets, making some mistakes in the top positions of the ranking, and it is near the baseline in the rest of the collection. On the other hand, the proportion of good pages per bucket for CS-NS is always better than TrustRank, successfully demoting the spam web pages into the last positions of the ranking. Our third approach, C-GS, also presents a good behavior for all the collection, avoiding a higher number of spam web pages than TrustRank in the first positions, except for the second bucket. In this last case, although the inclusion of metrics in the link-based algorithm has proved to be effective, the selection of the sources using specific thresholds seems to slightly penalize the accuracy of the method, causing an over-fitting of the technique because of relying too much on the content-based metrics.

TABLE 3. nDCG scores for each technique

	nDCG
TrustRank	0.738104
S-NS	0.875050
CS-NS	0.878753
C-GS	0.863445

Finally, as a summary of the different performances of each technique, we show in Table 3 the *nDCG* obtained by each method. This metric highlights the positions that spam web pages have achieved in the ranking computed by each technique.

Table 3 confirms that our three approaches outperform the results of TrustRank, achieving a better performance in terms of the demotion of spam web pages in the ranking. Again, the CS-NS approach obtains the best results in the experiments.

5. **Conclusions.** In this work, we have introduced a novel method, PolaritySpam, to deal with the web spam detection problem. It combines concepts from link-based and content-based techniques to avoid the negative effects of spam web pages in a web search engine. Our approach consists in a graph-based algorithm that uses a set of automatically chosen web pages as sources whose information will be propagated over the network. The information to be propagated is obtained by the aggregation of a set of content-based metrics, that are also used for the selection of the sources. Other methods for web spam detection, such as TrustRank, use similar ideas but the selection of the sources is a manual process which limits the number of sources that can be taken into account. Our system avoids this limitation without compromising the performance of the system, as it is shown in the experiments. Furthermore, the use of content-based metrics to include this information into the propagation algorithm has proved to be a very reliable method in the spam detection task.

On the other hand, PolaritySpam can be easily extended to include other content-based heuristics in the propagation algorithm, or to combine different methods for the selection of sources such as the inverse PageRank method proposed in [9]. These features can be useful to improve the performance of our technique, and also can make our system more flexible in terms of dealing with new possible spam strategies.

We plan to further our research by studying the relation between the improvement achieved by the inclusion of new heuristics and the time complexity of our algorithm. It is also interesting to find out the influence of the positive and negative source sets in the overall ranking of nodes, in order to be able to improve the source selection methods in different aspects, such as determining the minimum number of sources needed to obtain good results in the spam detection. We also intend to integrate our approach in a spam classifier, using many features in order to perform a binary classification of the web pages into spam or non-spam. Finally, we are very interested in applying these techniques in the detection of spam in social networks, a problem that is growing in parallel to the use of these on-line communities.

Acknowledgments. This work has been partially funded by the Spanish Ministry of Education and Science (HUM2007-66607-C04-04). Special thanks to the Terrier Team[3] of the Department of Computing Science of the University of Glasgow, and particularly to Dr. Craig Macdonald.

REFERENCES

[1] L. Becchetti, C. Castillo, D. Donato, S. Leonardi and R. Baeza-Yates, Link-based characterization and detection of web spam, *AIRWeb'06: Adversarial Information Retrieval on the Web*, 2006.

[2] A. A. Benczur, K. Csalogany, T. Sarlos, M. Uher and M. Uher, Spamrank – Fully automatic link spam detection, *Proc. of the 1st International Workshop on Adversarial Information Retrieval on the Web*, 2005.

[3] C. Castillo, D. Donato, L. Becchetti, P. Boldi, S. Leonardi, M. Santini and S. Vigna, A reference collection for web spam, *SIGIR Forum*, vol.40, no.2, pp.11-24, 2006.

[3] http://terrierteam.dcs.gla.ac.uk

[4] C. Castillo, D. Donato, A. Gionis, V. Murdock and F. Silvestri, Know your neighbors: Web spam detection using the web topology, *Proc. of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, pp.423-430, 2007.

[5] G. V. Cormack, M. D. Smucker and C. L. A. Clarke, Efficient and effective spam filtering and re-ranking for large web datasets, *Computing Research Repository*, 2010.

[6] L. da F. Costa, F. A. Rodrigues, G. Travieso and P. R. V. Boas, Characterization of complex networks: A survey of measurements, *Advances in Physics*, vol.56, no.1, pp.167-242, 2005.

[7] F. L. Cruz, J. A. Troyano, F. J. Ortega and F. Enríquez, Automatic expansion of feature-level opinion lexicons, *Proc. of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, Portland, OR, USA, pp.125-131, 2011.

[8] D. Fetterly, M. Manasse and M. Najork, Spam, damn spam, and statistics: Using statistical analysis to locate spam web pages, *Proc. of the 7th International Workshop on the Web and Databases*, New York, NY, USA, pp.1-6, 2004.

[9] Z. Gyongyi, H. Garcia-Molina and J. Pedersen, Combating web spam with trustrank, *Technical Report 2004-17*, Stanford InfoLab, 2004.

[10] H. Hama, T. T. Zin and P. Tin, A hybrid ranking of link and popularity for novel search engine, *International Journal of Innovative Computing, Information and Control*, vol.5, no.11(B), pp.4041-4049, 2009.

[11] T. H. Haveliwala, Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search, *Technical Report 2003-29*, 2003.

[12] M. R. Henzinger, R. Motwani and C. Silverstein, Challenges in web search engines, *SIGIR Forum*, vol.36, no.2, pp.11-22, 2002.

[13] G. Jeh and J. Widom, Simrank: A measure of structural-context similarity, *Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA, pp.538-543, 2002.

[14] J. M. Kleinberg, Authoritative sources in a hyperlinked environment, *Journal of the ACM*, vol.46, no.5, pp.604-632, 1999.

[15] P. Kolari, T. Finin and A. Joshi, Svms for the blogosphere: Blog identification and splog detection, *AAAI Spring Symposium on Computational Approaches to Analysing Weblogs*, 2006.

[16] V. Krishnan, Web spam detection with anti-trustrank, *ACM SIGIR Workshop on Adversarial Information Retrieval on the Web*, Seattle, Washington, USA, 2006.

[17] C.-C. Lai, C.-H. Wu and M.-C. Tsai, Feature selection using particle swarm optimization with application in spam filtering, *International Journal of Innovative Computing, Information and Control*, vol.5, no.2, pp.423-432, 2009.

[18] M. Marchiori, The quest for correct information on the web: Hyper search engines, *Computer Networks and ISDN Systems*, vol.29, no.8, pp.1225-1235, 1997.

[19] G. Mishne, D. Carmel and R. Lempel, Blocking blog spam with language model disagreement, *Proc. of the 1st International Workshop on Adversarial Information Retrieval on the Web*, 2005.

[20] M. A. Najork, H. Zaragoza and M. J. Taylor, Hits on the web: How does it compare? *Proc. of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York, NY, USA, pp.471-478, 2007.

[21] M. Najork, Web spam detection, *Encyclopedia of Database Systems*, pp.3520-3523, 2009.

[22] A. Ntoulas, M. Najork, M. Manasse and D. Fetterly, Detecting spam web pages through content analysis, *Proc. of the 15th International Conference on World Wide Web*, New York, NY, USA, pp.83-92, 2006.

[23] L. Page, S. Brin, R. Motwani and T. Winograd, The pagerank citation ranking: Bringing order to the web, *Technical Report*, 1999.

[24] P. Tin, T. T. Zin, T. Toriu and H. Hama, Reliability based web information ranking system, *Proc. of the 4th International Conference on Innovative Computing, Information and Control*, Kaohsiung, Taiwan, pp.294-297, 2009.

[25] B. Wu, V. Goel and B. D. Davison, Propagating trust and distrust to demote web spam, *Proc. of Models of Trust for the Web, A Workshop at the 15th International World Wide Web Conference*, Edinburgh, Scotland, 2006.